Delivering Services by Building and Running Virtual Organisations

Duong Nguyen, Simon Thompson

British Telecommunication, Adastral Park, Ipswich, UK

Jigar Patel, Luke W T Teacy, Nicholas R Jennings
Mike Luck, Viet Dang

School of Electronics and Computer Science, University of Southampton, UK

Stuart Chalmers, Nir Oren, Timothy J Norman
Alun Preece, Peter M D Gray

Department of Computing Science, University of Aberdeen, UK

Gareth Shercliff, Patrick J Stockreisser
Jianhua Shao, W Alex Gray, Nick J Fiddian

School of Computer Science, Cardiff University, UK

Abstract

In our view, customers in the future are likely to obtain their services from coalitions of service providers. These coalitions can be described as virtual organisations (VOs); they are group of service providers that form relationships to service customers' demands on an ad-hoc basis. For a VO to be effective, it must be reliable and scalable, and realistically, it must be created and maintained in a dynamic, open and competitive environment. The CONOISE-G project has focused on resolving the technology challenges that emerged from these requirements. Specifically, CONOISE-G provides mechanisms to assure effective operation of VOs in the face of failure, unexpected events and changing requirements in dynamic, open and competitive environment. In this paper, we describe the CONOISE-G system; motivated by a scenario based on mobile service provision; outline its use in the context of VO formation and perturbation and review current efforts to progress the work to deal with unreliable information sources. [1]

[1] Primary contact: Duong Nguyen, PP12 Orion 1, Adastral Park, Martlesham Heath, Ipswich IP5 3RE, United Kingdom
Email: duong.nguyen@bt.com
Telephone: +44 1473 605 894

1 Introduction

The engineering of systems using approaches that establish a fixed organisational structure is not sufficient to handle many of the issues inherent in large-scale open environments (in particular, the heterogeneity of the different actors, trust and accountability, failure handling and recovery, and societal change (1; 2)). This restriction is especially significant for companies such as BT, who are currently adopting Service Oriented Architectures (SOA) for their IT infrastructures and service provision platforms. SOA are networks of web services and other software components connected together using industry standard middleware and they are basic components of BT's 21CN vision (3). In addition, customers are increasingly demanding a high degree of customisation which adds to service complexity. In this paper we report on our attempts to develop a system that can address these requirements.

In more detail, VOs are composed of a number of software agents. A software agent is an computer program representing different individuals, departments and organisations that possess a range of problem-solving capabilities and resources. While such agents are typically self-interested, there are sometimes potential benefits to be obtained from pooling resources: either with a competitor (to form a coalition) or with an entity with complementary expertise (to offer a new type of service). This can be used as the cue for the formation of a VO in which distinct, autonomous agents come together to exploit a perceived niche. When this is successful, the collection of independent agents acts as a single conceptual unit in the context of the proposed service, requiring that the participants cooperate and coordinate their activities in delivering the services of this newly formed organisation.

This requires that the participants have the ability to manage the VO effectively. In dynamic environments, however, the context may change at any time, so that the VO may no longer be viable. It must then either disband or re-arrange itself into a new organisation that better fits the circumstances.

This paper describes technologies developed to address both these phases. VOs provide a way of abstracting the complexity of open systems to make them amenable to service development. The organisational structure, participant responsibilities, synchronisation concerns and economic mechanics of the VO are hidden from the VO customer. This has two benefits:

- Agents can be used to bridge between requester and providers to organise the VO and to provide a layer of flexibility between requesting services and the underlying infrastructure.

- The VO fulfils the role of information hiding in that the internal mechanics are abstracted away from the requesting service, and the VO formation and management system either supports a request or fails at well-defined points.

In the current generation of composed service applications, VOs are statically defined by developers using, for example, technologies such as WS-BPEL. This means that such applications are incapable of handling dynamic situations and reconfiguring themselves in an automated manner. Automated formation and ongoing management of VOs in open environments constitutes a major research challenge, a key objective of which is to ensure that they are both agile (can adapt to changing circumstances) and resilient (can achieve their aims in a dynamic and uncertain environment). In addition to constraints that relate to issues such as resource management and bidding strategies, we must also consider softer constraints relating to contract management, trust between VO participants and policing of contracts.

Against this background, the CONOISE-G project (Constraint-Oriented Negotiation in an Open Information Services Environment for the Grid [2]) is directed at addressing just these issues. The CONOISE-G vision is a set of system agents, operating at the application layer of the architecture, working together to support robust and resilient VO formation and operation. It aims to provide mechanisms to assure effective operation of agent-based VOs in a dynamic, open and competitive environments, where entities compete for limited resources and unexpected events must be considered. Furthermore, this abstraction facilitates the development of highly personalized applications.

Moreover, to operate an effective VO, we are required to monitor QoS levels (see section 3) and use this data to support mechanisms for recognising and addressing contract violations, once they have occured. Addressing these concerns is integral to the wide-scale acceptance of agent-based VOs. To that end, in this paper, we describe the CONOISE-G system, in which VO formation is grounded on three key technologies (4):

- Decision-making, including coalition formation.
- Trust and reputation management, including contract enforcement.
- Quality of service (QoS) assessment in the context of sparse short term interactions between entities.

In addition, CONOISE-G has made a contribution by the construction of a functional prototype for dynamic re-formation of VOs through the integration of several different techniques.

The structure of this paper is as follows. First, a motivating example that introduces the need for VO formation and operation is given in section 2. Sec-

[2] http://www.conoise.org

ond, the system architecture, elaborating the different aspects identified above in support of robust and resilient operation is described in section 3. Third, the implemented prototype that underlies the core of the current work in the project is reported in section 4 together with some experiences gathered during the implementation in section 5. Finally, a brief recap and some directions for future work in section 6 conclude this paper.

2 A Motivating Scenario - Lucy at the Olympics

In this section, we describe a scenario that shows the kind of application that demonstrates the utility of using CONOISE-G technology. Lucy visits London in 2012 for the Olympic Games, using her PDA to access various multimedia services (news, clips from the Games, tickets for events, text messaging, and *ad-hoc* entertainment opportunities, such as streaming video). Many service providers offer such services, so Lucy must determine available providers, select an optimal package, and then track the changing market for better deals.

This kind of decision is typically made by users on the basis of recommendations from friends, or because of the influence of marketing. Here we are presenting the CONOISE-G system as empowering Lucy to make a rational choice very quickly. The application allows her to specify her service choices, and the infrastructure will form the VO that will most closely match those requirements at the best balance of price, probable quality of service and probable reliability. Of course this is a complex task.

Suppose there are five service providers (`SP1`, ..., `SP5`), as in Table 1, each offering relevant multimedia services. These services form three groups: *video content* (Entertainment and Game Clips services), *HTML content* (News and Ticketing services) and *text messaging* (Text service). They can be requested individually or taken as a package, with the constraint that the two services offered by `SP2` must be taken together.

SP	Entertainment	News	Text	Games	Tickets
SP1	30	20			5
SP2		10	50		
SP3			100	30	5
SP4	30	10		60	
SP5			50	45	10

Table 1
Potential Service Providers

We assume that these providers may demand different prices for the same

service, depending on the number of units requested. For example, SP1 may offer 20 news updates per day at £30 per month, and 10 updates at £25 per month. Also, the quality of services may not be stable: SP4 may offer Games clips with a frame rate of no less than 24 frames per second, but actually provide a rate that drops below that level. Finally, not all service providers are trustworthy, and what they claim may not be what a requester will get: SP5 may advertise sought-after tickets that it does not possess, and orders for tickets through SP5 may not always be honoured.

Now, suppose that Lucy wishes to purchase the service package of Table 2. It should be clear from Table 1 that many different solutions are possible. For example, for 50 minutes of entertainment, both SP1 and SP4 must be used, but different compositions of the two services are possible, with different price, quality and degree of trust.

Service Required	Units Required
Entertainment	50 mins per month
News	10 updates per day
Text messages	100 per month
Game Clips	60 mins per day
Ticketing	10 alerts per day

Table 2
Example service package request

During VO formation, multiple service providers may offer broadly similar services, each described by multiple attributes including, for example, price, quality, reputation and delivery time. We need to determine how the relevant services for a given service request may be discovered and how an optimal package may be selected, based on the above attributes. After VO formation, when the services are being provided, the VO enters the operation stage. During VO operation, however, the services available may change over time: new services may become available or providers may alter the way in which existing services are offered. In addition, the services provided may be subject to fluctuations in their quality and, in some cases, a service provider may simply break the contract and stop providing the service, leaving the VO short of one service. So we need to monitor the performance of the members of a VO in terms of their trustworthiness, quality of service and conformance to contract, and to restructure the VO in light of perturbations so that the integrity and usefulness of the VO are maintained. Thus, a poorly performing service may be replaced, a contract-breaking service may be dropped, and a new user requirement may be accommodated.

Creating and then effectively managing a VO in this type of dynamic environment thus poses significant research challenges. In seeking to address them,

we have developed a system for dynamic formation and operation of VOs. In the following sections, we outline the system architecture and describe its key components.

3 The CONOISE-G Architecture

The CONOISE-G architecture comprises several different agents, including *system agents* and *service providers* (SPs), as shown in Figure 1. The system agents are those needed to achieve core system functionality for VO formation and operation, while the service providers are those involved in the VO itself. For simplicity, we omit the discussion of some specific components that perform basic functions, such as a Yellow Pages (YP) agent, since they add little to the elaboration of the issues to be discussed here. The system agents that are shown in Figure 1 are discussed in more detail in the relevant subsections that follow.

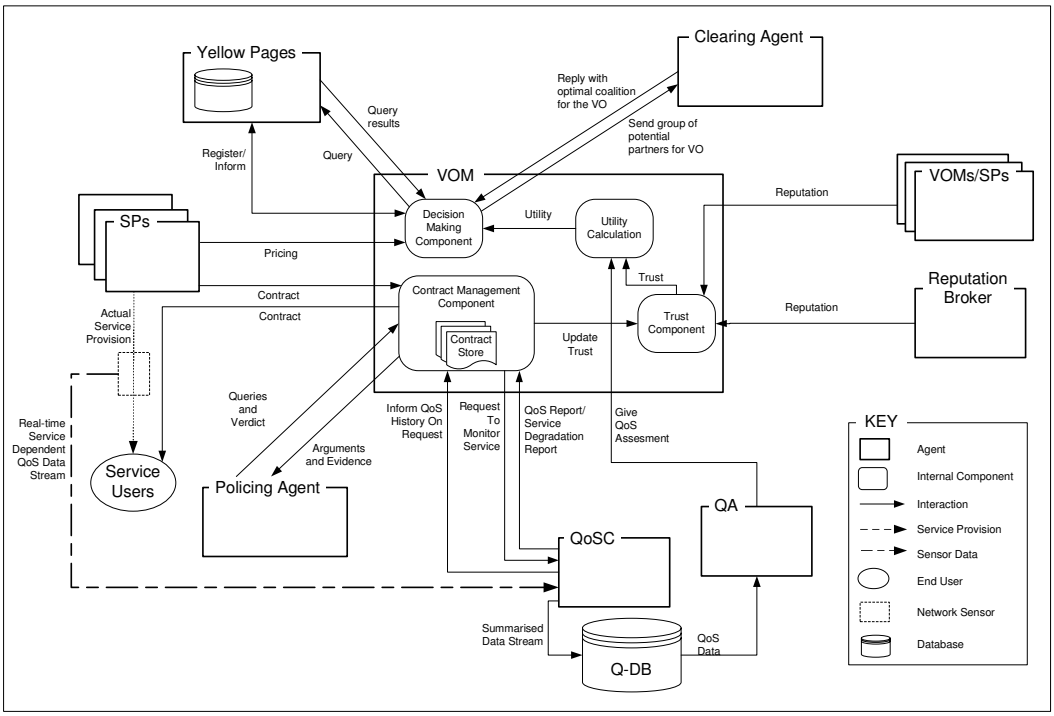

Fig. 1. The CONOISE-G system architecture

Assuming that service providers have already advertised their services to a YP, the VO formation process starts with a particular SP acting on behalf of a user, known as the Requester Agent (RA), which analyses the requester's service requirements, locates the relevant providers through the YP, and then invites the identified providers to bid for the requested services. The quality and trustworthiness of the received bids are assessed by the Quality Agent

(QA) and the trust component, respectively, and the outcome is combined with the price structure by a Clearing Agent (CA) (4) to determine which combination of the services/providers will form an optimal VO (in terms of price, quality and trust) for the requester. At this point, the VO is formed and the RA takes on the role of VO Manager (VOM), responsible for ensuring that each member of the VO provides its service according to contract.

During the operational phase of the VO, the VOM may request a QoS Consultant (QoSC) to monitor services provided by any members of the VO and any member of the VO may invoke a Policing Agent to investigate a potential dispute regarding service provision. Ultimately, our aim is to inform the user when the actual service level diverges from the agreed service level for the aggregate service previously specified. At present, however, policing is achieved on a per-component (service) basis.

When the QoS provision of a service (say the *news* service in the scenario) in the VO falls below an acceptable level, or some breach of contract is observed, the QoSC alerts the VOM, which initiates a VO re-formation process; the VOM passes relevant information (including service provider name and outcome of the contract held with that provider) to the trust component to ensure that the provider concerned is penalised to an appropriate level by updating its record of trust.

In this re-formation process, the VOM issues another message to the YP requesting a list of SPs that can provide the *news* service. As before, the YP identifies possible SPs, and bids are received and evaluated, resulting in the CA determining the best SP to replace the failed provider. At this point, the VOM re-forms the VO with the new SP replacing the old one, and instructs the QoSC to stop monitoring the old SP and to monitor the new one instead. We now proceed by discussing the core technical components of the architecture in more detail.

3.1 Decision Making in VO Formation

In developing a model of VO formation, there are a number of issues that must be taken into account including:

- An agent that is considering whether to join a VO must determine the conditions under which it is profitable for it to do so (see section 3.1.1).
- The agent that initiates the VO formation process must, given a number of offers, determine the best coalition it can create (see section 3.1.2).

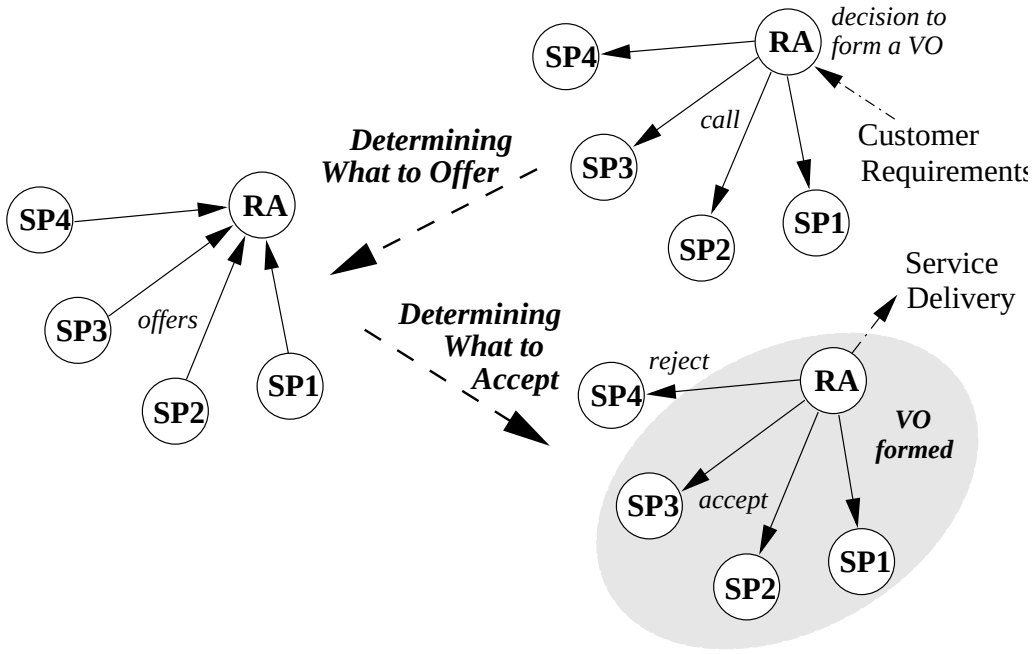

Fig. 2. The agent decision making process.

3.1.1 Determining What to Offer

The purpose of a service provider agent is to be able to create a bid in reply to a call for services, and decide how much resource it can, and more importantly, how much resource it *wants* to provide as a bid for the procurement of that service. Furthermore, any agent may, when considering what to offer, take on the role of the requester agent in figure 2 and issue a call for bids if it identifies a shortfall in its available resources. Each agent must, therefore, be able to act as a contractor and supplier in any given situation.

To give such dual-purpose functionality, we have designed a Constraint Satisfaction Program (CSP) that models the decision making process the agent must take in such scenarios.

Figure 3 shows one such scenario, where the agent acts as the supplier and receives a call for bids. It has the following possible responses: (i) it can decide *not* to bid for the service; (ii) it can bid using just its own resources; (iii) it can provide a bid from within an existing VO collaboration utilising the combined VO's resources; or (iv) it identifies a need for extra resources not available within the existing VO. We can see that the last option represents the scenario where the agent becomes the contractor, and itself begins the process of issuing a call for bids to other agents in the environment.

The technique used to provide the decision making process is based on a cumulative scheduling CSP (5). Usually, this is defined as the maximum allowable limit from a finite 'pool' of resource that *can* be used collectively by the agents

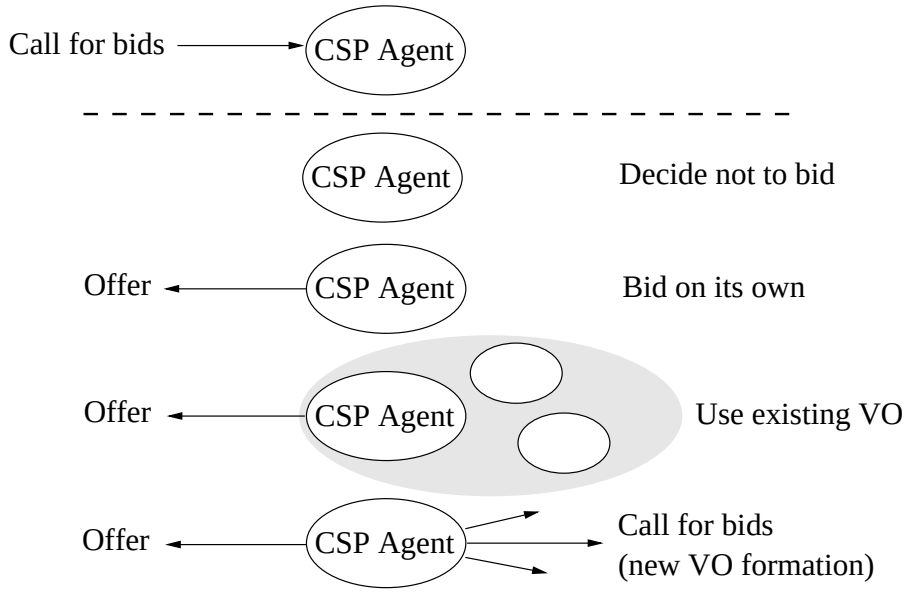

Fig. 3. The agent decision making process.

at any given time (6). We define our problem differently; rather than the agents taking resources from a communal resource, we have the agents contributing to the communal pool, and we define a *minimum* allowable limit so that the set of agents *must* provide this service *at least or above* the required threshold limit over the required time. If it is not possible, then we use the CSP to highlight the deficit and can then look to contracting-out for the provision of this shortfall.

To explain our cumulative scheduling based algorithm, we first define the problem. Given a set of n agents in a VO, each of whom can provide a specific finite amount of a resource $R \in \{R_1..R_n\}$, a set of start times describing when the agent can begin providing each of the resources $\{S_1..S_n\}$ and a set of durations over which the resource is available $\{D_1..D_n\}$ we can say, for an agent $i \in \{1 \ldots n\}$, that the function $\delta_i(t)$ evaluates to 1 if the current time t is within the agent's resource start and end time $(S_i < t \leq (S_i + D_i))$, and 0 otherwise. Then, an amount r of resource R is available over a time period $1..v$ iff $\forall t \in \{1..v\}\,(\sum_{i=1}^{n} R_i\delta_i(t)) \geq r$. In other words, the total sum of the resource provided by the set of agents with indices $\{1 \ldots n\}$ in a VO at any time between *1..t* does not fall below the resource limit r specified. Using this representation means that we can also use constraints on the agent resource domains to represent existing commitments on those resources.

In our scenario, this helps us to model the decision making process as the agent can look at the existing partners in its VO, as well as its own resources and the existing commitments, and see whether it can accommodate the new allocation of resources asked of it. As an example, let us look at an agent `a1` who is in a VO with two other agents `a2,a3`. All can provide a certain amount

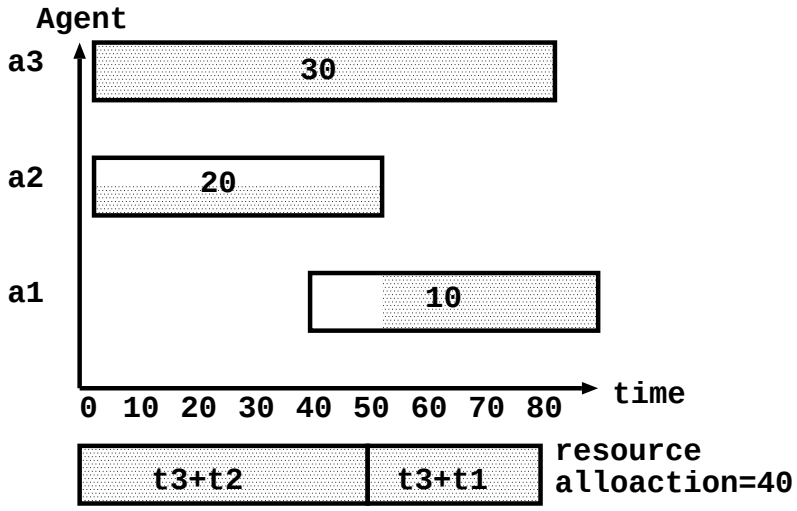

Fig. 4. An example schedule.

of bandwidth (10, 20 and 30 units respectively). Agent `a1` is asked to provide a total bandwidth amount of 40 units (as described in the introduction) from time 0 to 80, so it uses the knowledge of the amount of resources contributed from the other agents in the VO (along with its own) to work out if this is possible. Figure 4 shows an example allocation. A total rate of 40 units is provided by `a3` and `a2` between 0 and 50, then by `a3` and `a1` between 50 and 80. We can also add constraints on the resources available for each agent at each point in time to represent commitments under other contracts.

Of course there are many permutations that we can have in this resource allocation process. What we have described so far shows what the agent *can* do, but we also want to be able to model a utility that allows the agent to choose between competing viable allocations (i.e. decide what it *wants* to do).

We have implemented this utility using constraint reification, where each constraint on the domain of the resource has an associated value, 1 or 0, which depends on the success or failure of the constraint. For instance, using SICStus Prolog [3] notation, `X < Y# <=> B` states that if `X` is less than `Y`, the variable `B` is set to 1, otherwise it is set to 0. When the agents try to provide a new resource we take into account the current commitments of the agents (all the constraints currently posted against the resources) and we get a set of reified values for each commitment which we can then use to see which constraints are satisfiable alongside the new call for bids, and which ones 'fail', and so have a 0 value in their reification, that is, the resources cannot be allocated in the current situation. We can also highlight *where* the new bid is failing and identify the shortfall. Using this information, we also have a basis on which we can look at quality and pricing metrics (see section 3.4) for commitments

[3] The cumulative scheduling algorithm is implemented using the finite domain constraint library in SICStus (see http://www.sics.se/isl/sicstuswww/site/index.html).

in comparison to the new resource being bid for, and this therefore allows us to prioritise the commitments we have against any new ones that might arise. Before we discuss quality issues, however, we will address the problem of which offers the agent initiating VO formation should accept to create the best, or at least a satisfactory, VO.

3.1.2 Determining What to Accept

Since VOs do not have a rigid organisational framework, the selection of partners is one of the most important activities in the formation of the VO (7). However, there are three requirements that need to be met by this process:

(1) The most suitable set of partners from those that are available should be selected. In this context, most suitable means the ones with lowest price bids. Note that the *price* here does not just mean the monetary value of the bids but may be a combined rating value, calculated from monetary value and other attributes of the goods/services offered by the partners (e.g. delivery time).
(2) The selection should occur within a computationally reasonable time frame so that the market niche can be exploited as it becomes available.
(3) The potential partners should be able to vary their bid depending on their involvement in the VO. Thus, for example, a partner may be willing to complete services more cheaply if it has a high degree of involvement in the VO (because the intrinsic costs can be depreciated over many instances). In contrast, if a partner has a comparatively small involvement then the unit cost may be much higher.

Given the open nature of the environment and the lack of a pre-ordained structure, we believe this creation process is best achieved using some form of marketplace structure (auction). This is because markets are a highly effective structure for allocating resources in situations in which there are many self-interested and autonomous stake-holders. There are, however, many different types of auction (see (8) for a classification) but in this work it was decided to adopt a *combinatorial auction* approach.

In a combinatorial auction, bidders may bid for arbitrary combinations of items. For example, a single bid may be for 5 movies, 24 news updates (per day) and 20 minutes of phone at a total price p per month. A more complicated bid may be for q_1 movies and q_2 news updates at price $(30 * q_1 + 3 * q_2)$ if $q_1 < 10$ or $q_2 < 24$, and at price $(20 * q_1 + 2 * q_2)$ if $q_1 \geq 10$ and $q_2 \geq 24$. This particular type of auction is suitable for this problem because the degree of flexibility in expressing offers allows the potential partners to vary their bid depending on their involvement in the VO. However, the main disadvantages of combinatorial auctions stem from the lack of a compact and expressive

bid representation and efficient clearing algorithms for determining the prices, quantities and trading partners as a function of the bids made. Without such algorithms, because of the computational complexity of the problem, there may be unacceptable delays for auctions that have only a medium number of participants. Thus, in the CONOISE context, a compact and expressive bid representation language and efficient clearing algorithms for combinatorial auctions have been developed (9).

Specifically, we developed a bid presentation language where the price of a package, $P_i(r_1, \ldots, r_m)$ is specified as: $\omega_i(t_1, \ldots, t_m) \cdot (\sum_{j=1}^{m} P_i^j(r_j))$, where P_i^j is the price function of agent i for item j, in the form of a piecewise linear curve (i.e. the function's graph is composed of many segments, each of which is linear), t_j is the segment number of P_i^j that r_j belongs to and ω_i is a function that expresses correlations between items in the set of segments.

More precisely, each piece-wise linear function P_i^j is composed of N_i^j linear segments, numbered from 1 to N_i^j. Each individual segment with segment number l, $1 \leq l \leq N_i^j$, is described by a starting quantity $s_{i,l}^j$ and an ending quantity $e_{i,l}^j$, a unit price $\pi_{i,l}^j$ and a fixed price $c_{i,l}^j$, with the meaning that: bidder i wants to trade any r units of item j, $s_{i,l}^j \leq r \leq e_{i,l}^j$ with the price $P = \pi_{i,l}^j \cdot r + c_{i,l}^j$.

Note that the segments are not required to be continuous; that is, $(s_{i,l+1}^j - e_{i,l}^j)$ may not equal 1. Also, for convenience, we call segment number 0 the segment in which the starting quantity, the ending quantity, the unit price and the fixed price are all equal to 0. Thus, the number of segments of P_i^j, including this special segment, will equal $N_i^j + 1$.

Having developed a compact representation language as described above, two sets of clearing algorithms have been implemented. One algorithm has polynomial complexity and was shown to produce a solution within a finite bound of the optimal, while the other is not polynomial but is guaranteed to produce the optimal allocation (9). In particular, the former uses a greedy approach, and has a running time of $O(n^2)$, where n is the number of bidders. The solution it produces is shown to be within a finite bound of the optimal, which is proportional to n and K^{m-1}, where m is the number of items and K is a small constant. On the other hand, the latter is guaranteed to produce the optimal allocation, and has a worst-case running time that is proportional to $mn \cdot (K' + 1)^{mn}$, where K' is the upper bound on the number of segments of P_i^j. As these two sets of algorithms provide a trade-off between running time and optimality of solution, they provide the user with more flexibility. In cases where the running time is more crucial, the polynomial algorithms would be more appropriate, while in cases where optimality of the solution is more desirable, the optimal algorithms will be better suited.

Whenever interactions take place between different agents, the issues of trust and reputation become important. In particular, during the formation of a VO, we often have a choice of service providers to whom we may delegate tasks. In such cases, trust serves as an indicator of which of these possible partners are likely to carry out the task as specified, but its usefulness also extends into the other stages of the VO lifecycle.

In CONOISE-G, we take trust to be a particular level of the subjective probability with which an agent assesses that another agent will perform a particular action, both before it can monitor such an action and in a context in which it affects its own action (based on the definition from (10)). This probabilistic view of trust allows us to determine the *subjective probability* by considering the outcomes of previous encounters (known as direct interaction-based trust). However, in an open community it is likely that an agent will interact with many unknown entities with which it may not share an interaction history. In the absence of this shared history, the CONOISE-G trust system uses *reputation* information to establish the level of trust to place in another. Here reputation is defined as *a commonly held set of opinions about an entity* (11), and it is the aggregation of these common opinions that forms a level of trust. In more detail, the trust and reputation system (12; 13) consists of two distinct parts:

- The first is a trust component which is internal to all agents that require a trust metric in their decision-making process, as shown in Figure 1. Its function is to provide its owner agent with a level of trust for a given service and service provider, and it is insulated from the external environment by the agent that embodies it. As the agent interacts with others in the community, the outcomes of these interactions are stored in this component and are subsequently used to determine a trust value when required. In addition to calculating trust, the trust component calculates a level of *confidence* to be placed in that trust value. It is used by the trust component to reason about whether an agent itself has adequate evidence or whether it needs to obtain further (reputation) information from others.
- The second part of the trust system is a *reputation brokering* agent, several of which may serve as a distributed store of reputation information. These reputation brokers provide aggregated stores of trust information relating to specific service provider agents and each of their services. However, before any agent can query the broker, the broker must obtain the trust information that will form the query result. We achieve this using a *subscribe and publish* mechanism, by which the broker subscribes to agents in the community which then publish their internal information (the store of outcomes based on their individual direct experiences) to the broker.

3.3 Policing within a VO

While trust and reputation ratings can reduce the likelihood of poorly performing (or malicious) agents becoming part of a VO, they do not offer any mechanism for minimising the impact of undesirable behaviour, such as an agent contracting to provide services it does not deliver. Given this, the goal of the policing system is to determine whether a party is in breach of a contract, determine if any corrective action (as stipulated in the contract) should be taken, and inform the trust mechanism to allow sanctions to be imposed. Given the scalability concerns inherent in large, open distributed systems, the CONOISE-G system responds to reported exceptional circumstances, rather than monitoring all operations.

In more detail, the policing system initiates an investigation following the receipt of a complaint from a VO participant. The process begins by obtaining the relevant contract at the centre of the dispute, and gathering evidence to determine the actual state of affairs. This can take on a number of forms, including reports from agents in the system and other artifacts; it is recursive, in that one piece of evidence may have further evidence supporting or rebutting it. Furthermore, agents can submit evidence in support of, or against, a conclusion. The evidence gathered, therefore, constitutes a set of defeasible arguments in support of, and in defence of, the complaint. Given this, our approach borrows ideas from computational models of legal reasoning and legal argumentation (14).

In CONOISE-G, the representation of contracts is based on the emerging Web Services standard for agreements, WS-Agreement (15). However, due to the complexity of the environment, extensions to WS-Agreement are required, while we ignore some parts of the specification which are not useful in our domain.

3.4 QoS Assessment and Monitoring

In an open and dynamic service environment, QoS monitoring is characterised by the need to handle many sparse interactions over an extended period of time. In our model we consider assessment and monitoring to be part of a wider QoS lifecycle: the specification, assessment, monitoring and logging of QoS for service provision, supported by information flows between each stage and a QoS taxonomy for expressing user requirements.

In CONOISE-G we take a user centric approach in fully supporting the QoS lifecycle (16). Specifically, service providers are able to specify their QoS promises, and service users specify their QoS requirements using a DAML-

S derived QoS taxonomy (17). To allow for diverse services with potentially conflicting definitions of a particular QoS attribute, within this taxonomy we separate QoS attribute specifications from their measurement methods and allow mappings between the two to be established by individual services. That is, individual providers may define any QoS attributes in an expression they prefer for their services, but how such attributes will be monitored depends on which measurement methods the providers choose. This enhances the applicability and generality of our proposed framework.

In particular, three components are responsible for supporting the QoS lifecycle within the CONOISE-G architecture, as shown in Figure 1:

- First, during the operation of a VO the QoS Consultant (QoSC) monitors the performance of each of the delivered services against their promised QoS, so as to ensure that the VO as a whole is performing to agreed levels as well as each participating entity.
- Second, the streamed information that is generated from the QoS monitoring stage is summarised and stored within the QoS Database (QDB) component providing a permanent record of the performance of each service provider within the environment.
- Third, the QoS Assessment (QoSA) component provides for the effective handling of a user's QoS requirements at the service discovery stage and uses the QDB to establish the likelihood that a particular service provider will be able to meet those requirements. In our work, we extend current approaches to the incorporation of QoS Assessment in service discovery (2; 18). This aspect of our work is particularly relevant to the area of Grid computing, where dynamic, near instantaneous discovery, assessment and composition of resources is the norm.

4 The CONOISE-G Implementation

The CONOISE-G environment is FIPA [4] compliant and the implementation uses the JADE [5] agent platform. Agents communicate by exchanging FIPA ACL (agent communication language) messages, the content of which is defined using lightweight ontologies expressed in Semantic Web (SW) representations (following experience from previous work (19)). We chose these representations in preference to the more conventional use of FIPA-SL in the content of FIPA messages for a number of reasons. First, the SW representations are more widely used than FIPA-SL, so CONOISE-G is lent greater interoperability by aligning with W3C recommendations. Second, we can reuse

[4] `http://www.fipa.org`
[5] `http://sharon.cselt.it/projects/jade`

existing schemas and ontologies; for example, we borrowed heavily from the DAML-S service ontology. Thus, we would be in a position to exploit any existing schemas or ontologies in a particular application domain. Third, particularly at the lower (RDF) layers of the SW formalism stack, the semantics of the data model are much simpler than FIPA-SL (while still adequate for operational use), so there is less of a learning curve for designers and implementors of CONOISE-G agents (and much well-tested software for processing RDF, unlike FIPA-SL).

In the current system, we have created a set of interrelated ontologies expressed in a relatively lightweight manner as RDF schemas. For now, RDFS is sufficiently expressive to capture usable structures, and has allowed us to rapidly develop the necessary message formats for inter-agent communication in our scenario. We envisage the definitions in the ontologies being refined with the addition of OWL (Web Ontology Language) statements once the formats have stabilised through further testing and refinement. A sample RDF messag expressed using a number of the ontologies is shown in Figures 5. This is a sample call for bids, as issued to SPs. This consists of an instance of a user **Requirement** structure, stating a number of services that the user's requirement *consistsOf*, and also a *qualityPreference* property, indicating that the most important thing for this user is lowest cost. The descriptions of each required service are adorned with service-specific properties; for example, the **MovieContent** requirement specifies a number of movies (per month), a subscription preference, and a genre type. This illustrates the use of terms from three CONOISE-G ontologies:

- the `package` ontology describes service packages, defining terms such as the class **Requirement** and the property *consistsOf*;
- the `quality` ontology describes domain-independent quality-of-service terms such as the *qualityPreference* property, and its various settings such as "minCost";
- the `media` ontology defines all application domain-specific terms for the Olympics scenario, including the service classes **MovieContent**, **HtmlContent**, **PhoneCalls**, and **TextMessaging**, all of which the ontology defines to be (indirect) sub-classes of the generic CONOISE **ServiceProfile** class (closely based on DAML-S).

As can been seen, the capability to create modular, interlocking ontologies using the SW formalisms allow us to build up quite elaborate information representations, all of which are easily serialisable in a portable, open XML syntax, and easily parsed and processed using tools such as Jena2 [6].

In terms of the user interface, the GUI (in Figure 6) shows two large windows.

[6] `http://www.hpl.hp.com/semweb/jena2.htm`

```
<rdf:RDF
  xmlns:rdf=‘‘http://www.w3.org/1999/02/22-rdf-syntax-ns#’’
  xmlns:quality=‘‘http://conoise.org/ontologies/quality#’’
  xmlns:media=‘‘http://conoise.org/ontologies/media#’’
  xmlns:package=‘‘http://conoise.org/ontologies/package#’’>
  <package:Requirement rdf:about=‘‘http://conoise.org/samples/request’’>
    <quality:qualityPreference rdf:resource=
            ‘‘http://conoise.org/ontologies/quality#minCost’’/>
    <package:consistsOf
      rdf:type=‘‘http://conoise.org/ontologies/media#PhoneCalls’’
      media:numberOfMinutes=‘‘25’’/>
    <package:consistsOf>
      <media:MovieContent media:numberOfMovies=‘‘72’’>
        <media:subscriptionType rdf:resource=
            ‘‘http://conoise.org/ontologies/media#monthly’’/>
        <media:mediaStyle rdf:resource=
            ‘‘http://conoise.org/ontologies/media#scienceFiction’’/>
      </media:MovieContent>
    </package:consistsOf>
    <package:consistsOf>
      <media:HtmlContent media:updateFrequency=‘‘24’’>
        <media:mediaStyle rdf:resource=
          ‘‘http://conoise.org/ontologies/media#news’’/>
      </media:HtmlContent>
    </package:consistsOf>
    <package:consistsOf
      rdf:type=‘‘http://conoise.org/ontologies/media#TextMessaging’’
      media:numberOfMessages=‘‘100’’/>
  </package:Requirement>
</rdf:RDF>
```

Fig. 5. RDF call for bids sent to SPs

The window split into four columns (in the background) shows the requirements made by a user for a composite service, the registered SPs and their services, the bids that the SPs make in response to a VOM's request and the last column shows the SPs that make up the VO. When the VO is re-formed, the new SP is incorporated into the *fourth column* display. The window in the foreground simulates a service provision episode of a movie service on a PDA. Here, the monitored (simulated) quality of service providers is also represented. Currently, the GUI shows the QoS being provided by each software agent in a VO as a dynamically expanding line graph. The QoS of a VO is a function of its members' QoS in two ways: (i) if the agent responsible for a particular service is changed, then the QoS provided by the VO is a function of the new agent's performance, rather than the old agent's; (ii) the QoS of a VO may be a function not only of the performance of the agent responsible for that resource, but also of other agents in the VO that provide prerequisite resources.

5 The CONOISE-G experiences

During the CONOISE-G project, there were various difficulties that were encountered. In this section, we present the challenges we faced and what we

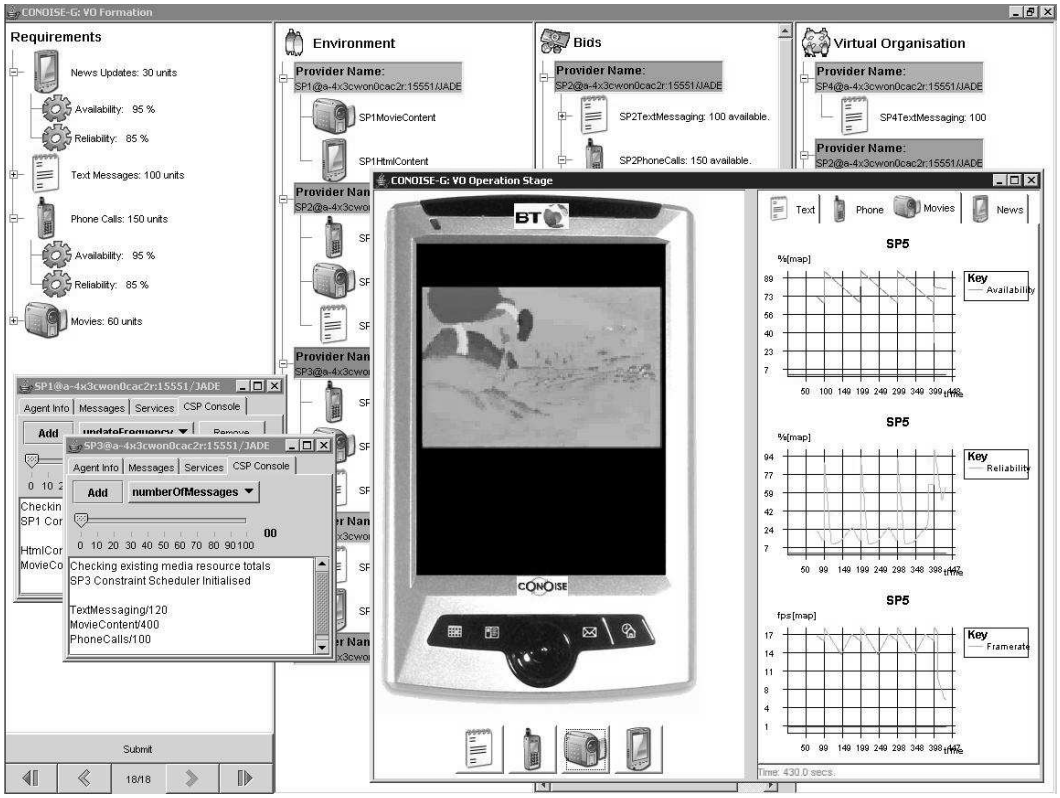

Fig. 6. The CONOISE-G user interface

learnt in addressing these challenges.

Early in the design process we were faced with an important decision, which was to decide whether we wanted to create the CONOISE demo using GRID software or FIPA standard agents. The GRID option would allow other to see the direct application of CONOISE technology to the GRID domain; however it also meant using, as a foundation for the demo, software that at the time was unstable and in a process of transition between releases. It was decide that instead of trying to create a new agent platform, which would have absorbed many man-hours from the CONOISE team, the demo would be created using open source, FIPA standard implementations. It was hoped at the time that significant functionalities from the selected platforms would be used in the CONOISE-G demo. However, in practice, the main benefit has been to provide a standard messaging infrastructure.

Another example of problems with the core CONOISE mechanisms were the issues related to the CSP mechanism that was used by the CONOISE service providers. After the first rudimentary version of the demo, the CSP mechanisms were implemented in Prolog, but had to be re-designed because of the multi-agent nature of the demo. To make the mechanism more efficient they had to be rewritten in Java. Additional problems with the CSP mechanism, caused by the need to run concurrent multiple CSPs in Prolog, were addressed

by redesigning the CSP using the same algorithm but using utility functions to model the reification, a feature that was only available in SICStus Prolog (see section 3.1).

6 Conclusions and Future Work

In this paper, we have described an agent architecture for re-forming VOs in the face of unreliable information, through the use of a range of techniques that support robust and resilient VO formation and operation for application to realistic scenarios. Specifically, the paper described our implemented prototype of the system (based on the original CONOISE work (4)), and highlighted the work being done on extending the system to incorporate more sophisticated application scenarios. The key problems encountered in the design phase of the system were those of standardising communication between agents and defining ontologies in a sufficiently detailed manner so as to allow agents to use them in their reasoning. These problems were overcome, wherever possible, by adopting standard technologies, such as RDF, FIPA and WS-Agreement. However, in other cases new technologies were needed, due to the change in application domain and technology since the CONOISE project. In the short term future it is important that the CONOISE-G mechanisms are made robust and resilient to failure (due to unforeseen circumstances and due to malicious intent). For this it will be necessary to develop strict policing mechanisms and evolve the trust mechanism to allow for representing the trust for a group and to enhance the ability to filter opinions given by liars.

It's not possible to provide a quantified evaluation of the business value of the CONOISE technology, after all this is research. However, it is clear that each of the technology challenges that we outlined in section 2 have been addressed head on by the techniques we have described. This will make it possible to implement applications that can take advantage of the open service environments (like the GRID, or possibly, exposure of BT's capabilities in the 21CN) to provide dynamic and optimal applications to particular users at particular times, and by doing this this technology has the potential to be the means by which the energy (money) which is required to make any rich ecosystem (service ecosystem) florish.

As we noted in the introduction of this paper, SOA are becoming the architectures of choice for service provisions in the telecommunication industry. At the same time, a wider variety value added services are becoming available. To exploit this opportunity, the technology presented in this paper has been developed to support seamless, efficient, secure and convenient service provision.

6.1 Acknowledgements

This paper is based on earlier work (20), which has been revised and extended. CONOISE-G is funded by the DTI and EPSRC through the Welsh e-Science Centre, in collaboration with the Research and Venture division of BT Group CTO. The research in this paper is also funded in part by the EPSRC Mohican Project (Reference no: GR/R32697/01).

References

[1] M. Luck, P. McBurney, C. Preist, A manifesto for agent technology: Towards next generation computing, Journal of Autonomous Agents and Multi-Agent Systems 9 (3) (2004) 203–252.

[2] E. M. Maximilien, M. P. Singh, Toward autonomic web services trust and selection, in: In ICSOC04: Proceedings of the 2nd international conference on Service oriented computing, 2004, pp. 212–221.

[3] C. J. Strang, Next generation systems architecture - the matrix, BT Technology Journal 23 (1) (2005) 55–68.

[4] T. J. Norman, A. Preece, S. Chalmers, N. R. Jennings, M. Luck, V. D. Dang, T. D. Nguyen, V. Deora, J. Shao, A. Gray, N. Fiddian, Conoise: Agent-based formation of virtual organisations, in: Proceedings of the twenty-third Annual International Conference of the British Computer Society's Specialist Group on Artificial Intelligence (SGAI), Cambridge, UK, 2003, pp. 353–366.

[5] Y. Caseau, F. Laburthe, Cumulative scheduling with task intervals, in: In Logic Programing Proceedings of the 1996 Joint International Conference and Syposium on Logic Programming, 1996, pp. 363–377.

[6] P. Baptiste, C. Le Pape, W. Nuijten, Constraint-based scheduling: Applying constraint programming to scheduling problems, International Series in Operations Research and Management Science 39.

[7] S. A. Petersen, M. Gruninger, An agent-based model to support the formation of virtual enterprises, in: International ICSC Symposium on Mobile Agents and Multi-agents in Virtual Organisations and E-Commerce, 2000.

[8] P. R. Wurman, M. P. Wellman, W. E. Walsh, A parametrization of the auction design space, Games and Economic Behavior 35 (2001) 304–338. URL `citeseer.nj.nec.com/wurman00parametrization.html`

[9] V. D. Dang, N. R. Jennings, Optimal clearing algorithms for multi-unit single item and multi-unit combinatorial auctions with demand/supply function bidding, in: Proceedings of the Fifth International Conference on Electronic Commerce, 2003, pp. 25–30.

[10] D. Gambetta, Can we trust trust?, in: Trust: Making and Breaking Cooperative Relations, chapter 13, Basil Blackwell, 1988, pp. 213 – 237.

[11] J. Sabater, C. Sierra, Regret: A reputation model for gregarious societies, in: In Fourth Workshop on Deception Fraud and Trust in Agent Societies, 2001, pp. 61–70.

[12] J. Patel, W. T. L. Teacy, N. R. Jennings, M. Luck, Travos: Trust and reputation in the context of inaccurate information sources, Journal of Autonomous Agents and Multi-Agent Systems (2006) to appear.

[13] W. T. L. Teacy, J. Patel, N. R. Jennings, M. Luck, Coping with inaccurate reputation sources: Experimental analysis of a probabilistic trust model, in: In AAMAS 05: Proceedings of 4th International Joint Conference on Autonomous Agents and Multiagent Systems, 2005, pp. 997–1004.

[14] T. Bench-Capon, J. B. Freeman, H. Hohmann, H. Prakken, Computational models, argumentation theories and legal practice, in: C. A. Reed, T. J. Norman (Eds.), Argumentation Machines: New Frontiers in Argument and Computation, Kluwer, 2003, pp. 85–120.

[15] K. Czajkowski, Dan, A., J. Rofrano, S. Tuecke, M. Xu, Ws-agreement: Agreementbased grid service management, in: In Global Grid Forum, 2003.

[16] G. Shercliff, P. Stockreisser, J. Shao, W. Gray, N. Fiddian, Supporting qos assessment and monitoring in virtual organisations, in: In Proceedings IEEE International Conference on Services Computing, 2005, pp. 249–250.

[17] V. Deora, J. Shao, G. Shercliff, P. J. Stockreisser, W. A. Gray, N. J. Fiddian, Incorporating qos specifications in service discovery, in: In Proceedings of Second International Web Services Quality Workshop (WQW 2004), 2004, pp. 252–263.

[18] M. Tian, A. Gramm, H. Ritter, J. Schiller, Efficient selection and monitoring of qos-aware web services with the ws-qos framework, in: In Web Intelligence, 2004, pp. 152–158.

[19] G. Grimnes, S. Chalmers, P. Edwards, A. Preece, Granitenights a multiagent visit scheduler utilising semantic web technology, in: In 7th InternationalWorkshop on Cooperative Information Agents, 2003, pp. 137 – 151.

[20] J. Patel, W. T. L. Teacy, N. R. Jennings, M. Luck, S. Chalmers, N. Oren, T. J. Norman, A. Preece, J. Shao, A. Gray, N. Fiddian, S. Thompson, Agent-based virtual organisations for the grid, International Journal of Multiagent and Grid Systems 1 (4) (2005) to appear.